Efficacy of Bidirectional LSTM Model for Network-Based Anomaly Detection

Toya Acharya[1], Annamalai Annamalai[2], Mohamed F Chouikha[3]
Electrical and Computer Engineering
tacharya@pvamu.edu[1], aaannamalai@pvamu.edu[2], mfchouikha@pvamu.edu[3]
Prairie View A&M University, Prairie View, Texas

Abstract- The Internet is vital in daily applications such as education, health, business, etc. Increasing the usage of the Internet and technology also increases the risk. Cyber attackers can use technology to compromise the triad of the CIA (confidentiality, integrity, and confidentiality). Malicious activities occur in our surroundings without our knowing it. Cyberattacks cannot be seen physically, though occurring to our Internet of things (IoT) devices, personal computers, laptops, and even our networking devices. Network anomaly detection is an efficient way of detecting malicious activities. Network-based anomaly detection captures and analyzes attributes of abnormal behavior in a network. Machine learning and deep learning-based approach are attractive among various known methods for network anomaly detection because they can efficiently analyze big network traffic data for malicious activities and detect zero-day attacks. A Recurrent Neural Network (RNN) model is designed to recognize the sequential characteristics of data and then use the patterns to predict the coming scenario. In this research work, seven different optimizers (Nadam, Adam, RMSprop, Adamax, SGD, Adagrad, and Ftrl), epochs, batch size, and the ratio of training testing data size are analyzed for the Bidirectional Long Short Term Memory (Bi-LSTM) network anomaly detection which provides the highest anomaly detection accuracy of 98.52% on the NSL-KDD binary dataset. The performance is compared using accuracy and F1-score metrics. Performance assessment regarding the accuracy and F1-score revealed that the proposed Bi-LSTM anomaly detection model exhibited better performance than the other existing anomaly detection methods.

Keywords— Network Intrusion Detection System, Machine Learning, Deep Learning, LSTM, Bi-LSTM, NSL-KDD

I. INTRODUCTION

With the invention of information and technology, the most crucial information is transmitted in the form of bits from source to destination. The transmitted information can be voice, image, or data, containing banking information, personal information, network traffic, etc. Various tools or methods are available to detect and prevent intruders. Anomaly is a pattern in the dataset that does not fit into the usual behavior of the data, and some detection techniques are required to detect it. Outliers and anomalies are sometimes used interchangeably in the field of anomaly detection. Anomaly detection has numerous applications, including business, network intrusion detection, health monitoring systems, credit card fraud detection, and fault detection in critical information systems. Anomaly detection is important in cyber security for achieving solid protection against cyber adversaries. A system is considered secure if the three computer security principles of Confidentiality, Integrity, and Availability (CIA) are properly met [1]. An intrusion detection system is a method for monitoring and examining what is happening in a computer or network system to detect potential risks by evaluating how often CIA computer security guidelines are broken.

There are two categories of intrusion detection methods: signature-based intrusion detection systems (SIDS) and anomaly-based intrusion detection systems (AIDS). Anomaly detection systems are classified into two categories based on the sources: network-based and host-based intrusion detection systems. Anomaly detection techniques utilize labels to identify whether the data is normal or anomalous. There are three different anomaly detection techniques such as supervised, unsupervised, and semi-supervised anomaly detection methods. AIDS overcomes the SIDS's drawbacks by modeling normal behaviors using machine learning, statistical-based, or knowledge-based methods. The different anomaly detection approaches are listed below in Fig. 1 [2].

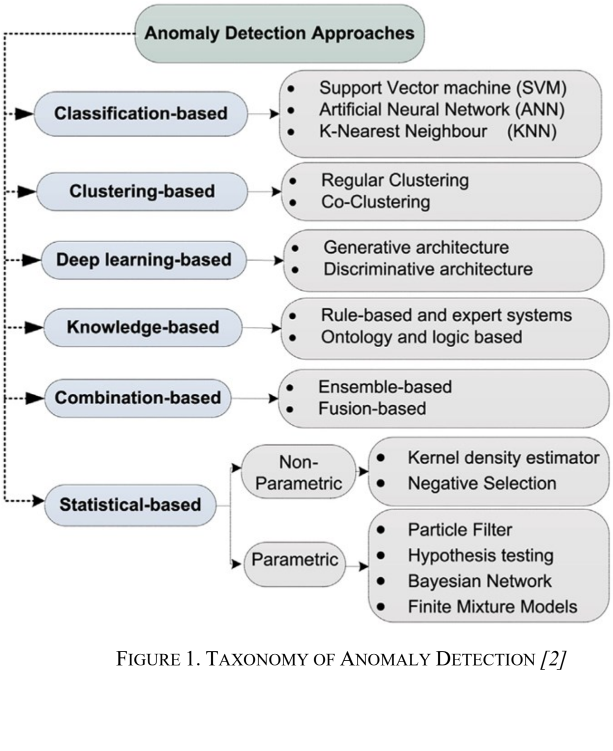

FIGURE 1. TAXONOMY OF ANOMALY DETECTION *[2]*

Deep learning can extract better representations for creating efficient anomaly detection models. The traditional machine learning-based network anomaly detection algorithms are more suited for small datasets and are mostly performance dependent on how the feature engineering is implemented. The split ratio is one of the dominant elements influencing the performance of traditional machine learning-based anomaly detection methods. The traditional ML methods are simple and have low resource consumption. Still, for huge datasets and large features, poorly performed and traditional ML cannot be worked on computer vision, natural language processing, image translations, etc. The Convolutional Neural Network (CNN) is mostly used in image datasets where the lower layer's neurons do the feature reduction in the network, usually identifying important small-scale features, such as boundaries, corners, and intensity differences. Then in higher layers, the network combines the lower-level features to form more complex features such as simple shapes, forms, and partial objects. And on the final layer, the network combines the lower features to produce the output or classification results. LSTM works differently than a CNN because an LSTM is usually used to process and make predictions given data sequences. RNNs were designed to retain long-range information so that the information is remembered and not lost in a long sequence. BiLSTM adds one more LSTM layer, reversing the information flow direction and overcoming the vanishing gradient problems.

The deep learning method overcomes the problems in traditional ML, such as being suited for huge datasets and large numbers of features. The performance of the deep learning-based anomaly detection algorithm depends on the number of neurons, number of hidden layers, types of activation function, number of samples (batch size), and epochs (iterations) during DL model training and testing. Selecting those hyperparameters, training testing data ratio, and architecture of neural network in deep neural networks is vital in increasing the detection accuracy of network anomaly detection systems.

II. Related Work

The volume of big data is growing daily, so the traditional machine learning algorithm cannot be performed well and needs intensive feature engineering tasks. Deep learning greatly improves detection performance. Still, the nature of the dataset used in network anomaly detection (balanced and unbalanced), the hyperparameters on deep neural networks, training, testing data size, and neural network architecture play a vital role in detecting the anomaly.

Authors [3] implemented the Bi-LSTM model to overcome the extensive feature engineering task required for traditional machine learning-based anomaly detection. Also, data augmentation used during data preprocessing on rare attacks (U2R, R2L) was applied to create the balanced NSL-KDD dataset resulting in the accuracy and F1 scores better than other comparison methods, reaching 90.73% and 89.65%, respectively. Authors [4] proposed a network intrusion detection algorithm that combined hybrid sampling with the deep hierarchical network where SMOTE was used to create the balanced dataset. The CNN-based Bi-LSTM hybrid technique was used to detect the anomalies on the NSL-KDD and UNSW-NB15 datasets and found the highest accuracy of 83.58% and 77.16%, respectively. The authors [5] proposed a Bi-directional GAN-based approach to the NSL-KDD and CIC-DDoS2019 datasets. The bidirectional GAN model works perfectly on the imbalance NSL-KDD dataset resulting in an accuracy of 91.12% and an f1 score of 92.68%. The authors used the GAN algorithm to improve the performance of the imbalanced NSL-KDD data. Authors [6] proposed a novel solution based on ACGAN and ACGAN-SVM to solve the data imbalance problem using generative adversarial networks to synthesize the attack traffic for IDS. The synthesized attacks are mixed with the original data to form the augmented dataset. The authors performed experiments on the NSL-KDD, UNSW-NB15, CICIDS2017, and RAWDATA datasets. Among the SVM, DT, and RF models, DT provides a higher F1-score of 92% on the NSL-KDD augmented dataset. During this work [7], the Authors used a Heterogeneous Ensemble Assisted Machine Learning Model for Binary and Multi-Class Network Intrusion Detection to overcome the data imbalance problem on KDD99, NSL-KDD, and UNSW-NB15 datasets. The model provides the 94.5% true positive rate and 96.2% AUC on the NSL-KDD dataset. Authors [8] concluded from the experimental results that the machine learning classifier's performance improved when the number of target classes decreased. Authors examined this concept on traditional machine learning models, including NB, J48, RF, BayesinNet, Bagging, and Adaboost on three NIDS datasets: UNSW-NB15, CIC-IDS2017_Thrusday, and KDD99.

Authors [9] studied the Recurrent Neural Network-based IDS model's performance in binary and multiclass classification. The number of neurons and different learning rates influences the proposed model's performance on the NSL-KDD dataset. The experimental results show that RNN-IDS is suitable for modeling a classification model with high accuracy. Its performance is superior to traditional machine learning (J48, artificial neural network, random forest, and support vector machine) classification methods in binary and multiclass classification. In this paper [10], the Authors propose a Convolutional Autoencoder-based (CAE) network anomaly detection method and found a detection accuracy of 96.87% on the NSL-KDD dataset. The CAE method was used to reduce and select the more relevant features for the anomaly detection algorithm. In this paper [11], the Authors explored the effectiveness of various Autoencoders in detecting network intrusions. The authors compared the performance of 4 different autoencoders, including Sparse Autoencoders, Undercomplete Deep Autoencoders, and Denoising Autoencoder, on the NLS-KDD dataset and achieved an accuracy of 89.34% by using a Sparse Deep Denoising Autoencoder. Authors [12] proposed a 5-layer autoencoder (AE)-based model better suited for network anomaly detection. The optimal model architectures are better equipped for feature learning and dimension reduction to produce better detection accuracy and f1-score by achieving the detection accuracy and f1-

score at 90.61% and 92.26%, respectively, on the NSL-KDD dataset. The authors utilized the reconstruction error function to decide whether a network traffic sample is normal or abnormal.

In this paper [13], the Authors implemented a network intrusion detection method combining CNN and Bi-LSTM network on the KDD99 dataset. The authors studied the effect of hidden layers, nodes, and the number of iterations to improve anomaly detection accuracy, where the accuracy of KNN, J48, Deep Forest, Naïve Bayes, Random Forest, and CNN-based Bi-LSTM. The CNN-based Bi-LSTM provides the highest detection accuracy of 95.4%. Authors [14] compared the single-layer and multilayer LSTM (4 layers) for weather forecasting on the weather dataset collected by Weather Underground at Hang Nadim Indonesia Airport with the highest validation accuracy of 80.60%. The different numbers of nodes on four hidden layers were used 200, 100, 90, and 50, and the data split ratio taken is 30 % test data for 500 epochs. The Authors [15] implemented the deep learning model based on Bi-directional LSTM on KDDCUP-99 and UNSW-NB15 datasets with outstanding results with 99% accuracy for both KDDCUP-99 and UNSW-NB15 datasets. Most existing models cannot efficiently detect rare attack types, especially User-to-Root (U2R) and Remote-to-Local (R2L) attacks. These two attacks often have lower detection accuracy than other kinds of attacks. Authors in [16] proposed a Bidirectional Long-Short-Term-Memory (Bi-LSTM) based intrusion detection system to handle the aforementioned challenges on the NSL-KDD dataset. This Bi-LSTM model provided an accuracy of 94.26% for binary classification.

The impact of batch size on the performance of CNN and the impact of learning rates were studied for image classification, specifically for medical images [17]. According to their findings, a larger batch size typically does not result in high accuracy, and both the learning rate and the optimizer employed will have a big impact. The network will train more effectively, particularly during fine-tuning, if the learning rate and batch size are reduced.

Various methods were implemented to overcome the data imbalance problem, including data augmentation on [3], SMOTE on [4], GAN technology on [5] [6], Heterogeneous ensemble assisted on [7], reducing the target class combining the smaller class in another new class on [8]. Huge numbers of research works related to network anomaly detection are examined in deep learning, including RNNIDS in [9], CAE in [10], Autoencoder in [11], multilayer Autoencoder in [12], CNN Bi-LSTM hybrid method in [13], and Bi-LSTM in [15] [16].

The Authors [16] and [15] do not mention the data preprocessing, train-test data ratio, and how those Bi-LSTM hyperparameters are adopted during their experiments. The authors [14] found Bi-LSTM for weather forecasting without referencing the values of the hyperparameters in their experiments. The authors [9] did not analyze the number of epochs and did not mention the percentages of the split ratio for the KDDTrain+ dataset. Most of the above research works are focused on the increase the model accuracy of either traditional or deep machine learning models. The research on selecting the hyperparameters in deep learning-based models, training testing data ratio, and architecture of deep neural networks are not focused on. Some researchers do not mention how those values are adopted in their research works. Hence, our research focused on improving those limitations on network anomaly detection systems by experimenting with the NSL-KDD dataset.

The main contributions of this research work are:

1) Investigating the effect of optimizers, batch size, and the number of epochs Vs performance of the Bi-LSTM.
2) Investing in the train and test split ratio to improve the network anomaly detection accuracy on the NSL-KDD.
3) Investing the number of layers and memory elements to improve the Bi-LSTM on the NSL-KDD dataset.
4) This research presents the development and implementation of network anomaly detection using a Bi-LSTM-based RNN model that can detect anomalies in a network with high accuracy of 98.52%.

III. System Model

The overall proposed model encompasses the following steps.

Step-1 Data Collection and Modelling
Step-2 Data Pre-procession
Step-3 Prepare the training and testing dataset
Step-4 Train and Test the Bi-LSTM Model
Step-5 Model Evaluation and anomaly detection
Step-6 Model Compare and Decision

The overall implementation schematic of the Bidirectional LSTM-based model is given in Fig. 2. A detailed discussion of the above-stated methods is provided in the subsequent sections.

1. *Data Collection and Modelling*

In this research, we used the KDDTrain+ dataset, one of the datasets available on NSL-KDD. This dataset contains the full NSL-KDD train set, including attack-type labels and difficulty level. It has 41 features with five distinct attack classes, Normal, DoS, Probe, R2L, and U2R. NSL-KDD [18] is an improved version of the KDD99 network intrusion dataset, does not include redundant records in the train set, and has no duplicate records in the test sets. The KDDTrain+ dataset contains 125973 records and 41 features. This dataset is balanced because 53.46% of records are normal, and 46.54% are abnormal. We selected this dataset because the normal and abnormal records contained the subset of the dataset is balanced.

2. *Data Pre-processing*

The KDDTrain+ dataset contains 125973 records and 41 features. During the data pre-processing, the class label is assigned 1 for normal and 0 for abnormal records; hence the dataset becomes the binary class dataset. Then, three categorical features: ‘protocol_type,’ ‘service,’ and ‘flag,’ are converted into numeric features using dummy one hot encoding. The standard

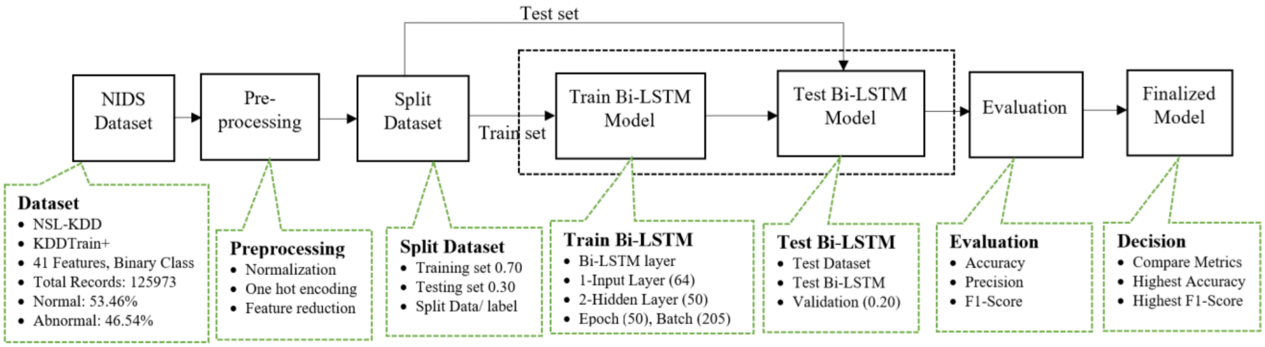

FIGURE 2. BLOCK DIAGRAM OF BI-LSTM-BASED MODEL

scalar method is used to normalize the dataset. For the feature reduction, attributes with more than 0.5 correlation with encoded attack label attribute are only preserved.

3. *Prepare the Training and Testing Dataset*

The train-test split approach measures how well machine learning algorithms perform when used to make predictions from data that was not used to train the model. Since the dataset we pre-processed is only one set of data, the two set of datasets to implement the machine learning algorithms. The train test split ratio does not have rules the researcher to follow but the common slit ratio are train 80% and test 20%, train 60% and test 40%, train 70% and test 30%, train 75% and test 25%. We performed the experiment to choose the split ratio where our Bi-LSTM model provides the best result is 70% train and 30% test dataset.

4. *Bi-LSTM Model*

A recurrent neural network (RNN) consists of feedback loops that process the sequences of data patterns and predict outcomes. Those loops allow the data to be shared with available nodes and predictions according to the collected information called memory. RNN has been used to solve machine learning problems such as speech recognition, language processing, and image classification. LSTM addresses the problem of the vanishing gradients of RNN. LSTM architecture consists of the memory block and three multiplicative units- the input, output, and forget gates which are analogous to write, read and reset operations for the cells. The LSTM memory cells can store and access data for extended periods because of the multiplicative gates, which prevents the vanishing gradient problem.

Conventional RNNs have the limitation that they can only use the previous context. Bidirectional RNN overcomes those problems by processing the data in both directions with two hidden layers, then feeding forwards to the same output layer. Generally, in a normal LSTM network, the output is taken directly. In the case of a bidirectional LSTM network, the output of the forward and backward layers at each stage is given to the activation layer. This output contains information on past and future patterns or data. The bi-directional LSTM predicts or tags the sequence of each element by using finite sequences in the context of previous and subsequent items. This results from two LSTMs processed in series, one from right to left and the other from left to right.

5. *Model Evaluation and Anomaly Detection*

Different experiments are performed to evaluate the Bi-LSTM model. Machine learning (ML) or deep learning (DL) model does not provide consistency in performance. Hence the model hyper-parameters need to be examined to obtain better performance. The determination of optimizer, the number of epochs, batch size, and train-test split ratio are determined by comparing the accuracy and F1-score of the Bi-LSTM model. Finally, the Bi-LSTM model performance parameters are compared with the previously published research results to evaluate our Bi-LSTM model's performance.

6. *Model Comparision and Decision Making*

Different sets of experiments to determine the values of the hyperparameters for the best result. The determination of optimizer, the number of epochs, batch size, and train-test split ratio are determined by comparing the accuracy and F1-score of the Bi-LSTM model. Finally, the Bi-LSTM model performance parameters are compared with the previously published research results to evaluate our Bi-LSTM model's performance. The performance metrics are recorded and compared for NSL-KDD binary NIDS datasets regarding f1-score and accuracy.

IV. RESULTS AND DISCUSSIONS

The experiments were adapted on a 64-bit Windows 10 machine with 16G RAM and an i7-1.99GHz processor. The versions of python, Keras, and TensorFlow used during this research work were 3.7.13, 2.6.0, and 2.9.1, respectively. The determination of training and testing data ratio, epochs, batch size, and selection of optimizer for the Bi-LSTM model was examined in the different experiments, which are explained below.

A. *Experiment-1 Optimizers Vs Bi-LSTM Model Accuracy*

In this experiment, the Bi-LSTM model experimented with the NSL-KDD dataset, whose specifications are given in the previous sections. The right optimizer is necessary for the model to improve training speed and performance. The selection of an optimizer is very important because it helps the ML /DL model to get results faster. TensorFlow supports nine optimizer classes, including Adadelta, Adagrad, Adam, Adamax, Ftrl, Nadam, RMSprop, SGD, and gradient descent were compared.

During this experiment, Bi-LSTM hyperparameters were chosen randomly, which are shown in Table 1 below. The Bi-LSTM model was created using 64 units, two bidirectional LSTM hidden layers with 50 units, and one output-dense layer. Each layer in Bi-LSTM used a relu activation function and a 20% dropout rate.

TABLE 1. OPTIMIZER VS. ACCURACY

Test size = 50%, epochs = 105, batch Size = 200			
SN	**Optimizer**	**Accuracy %**	**F1-Score %**
1	**Nadam**	**98.35**	**98.47**
2	Adam	98.33	98.44
3	RMSprop	98.28	98.39
4	Adamax	98.07	98.21
5	SGD	96.79	97.03
6	Adagrad	91.65	92.49
7	Ftrl	53.36	69.59

Observing the above results (Table 1), it can easily be found that Nthe adam optimizer is the winning optimizer with the highest accuracy of 98.35% and the highest f1-score of 98.47%. Nadam is an improved version of the Adam algorithm that integrates Nesterov momentum, improving the optimization algorithm's performance.

B. Experiment-2 Train Test Ratio Vs. Accuracy

The train-test split ratio and Bi-LSTM model accuracy were studied in this experiment. Data splitting is crucial in data science, especially when building models from data. The train-test split approach is used to quantify how well machine learning algorithms perform when used to predict outcomes from data that was not used to train the model. After the training is completed, the testing data set is utilized. There is no set guideline for how the data should be split on training and test data from the given data set. The test split ratio is examined to obtain better network anomaly detection using Nadam optimizer on the NSL-KDD binary dataset.

TABLE 2.TRAIN TEST RATIO VS. ACCURACY

Optimizer = Nadam, epochs = 105, batch Size = 200			
SN	**Test Data size %**	**Accuracy %**	**F1-Score %**
1	**30**	**98.48**	**98.57**
2	25	98.47	98.57
3	50	98.39	98.5
4	40	98.35	98.46
5	20	98.33	98.44
6	60	98.28	98.4
7	10	98.17	98.29
8	70	98.15	98.29
9	80	98.15	98.29
10	90	97.98	98.13

This experiment provides the train-test ratio for the highest network anomaly detection for the Bi-LSTM model on the NSL-KDD dataset. The performance metrics are recorded in Table 2, where the test split of 30% achieved the purposed Bi-LSTM model with the highest accuracy and f1-score of 98.48% and 98.57%, respectively.

C. Experiment-3 Batch Size Vs. Bi-LSTM Accuracy

The effect of the batch sizes on the Bi-LSTM accuracy and the training time was studied during this experiment. This experiment aims to find the optimal batch size for the best model performance.

TABLE 3. BATCH SIZE VS. ACCURACY

Optimizer = Nadam, epochs = 105, test_size= 0.30				
SN	**Batch Size**	**Accuracy %**	**F1-Score %**	**Prgm Exe time (sec)**
1	**50**	**98.48**	**98.58**	2127.2346
2	100	98.46	98.56	1228.779
3	15	98.45	98.56	5671.738
4	200	98.45	98.55	796.8976
5	300	98.45	98.55	553.4444
6	150	98.42	98.52	858.07
7	450	98.41	98.51	454.989
8	350	98.4	98.51	527.1532
9	400	98.38	98.48	460.8835
10	500	98.36	98.47	514.7698
11	250	98.35	98.46	616.4657

The smaller batch size introduces small amounts of data samples and takes longer to train the Bi-LSTM model compared to the larger batch size. Model accuracy, F1-score, is shown in Table 3. The experimental result shows that the batch size of 50 during this Bi-LSTM model for the NSL-KDD dataset produces the best results in terms of accuracy and f1-score. Larger batch sizes take less time to train but are less accurate, which is an important trade-off for this Bi-LSTM model.

D. *Experiment-4 Epochs Vs. Bi-LSTM Accuracy*

The number of times the learning algorithm will go over the complete training dataset is determined by the hyperparameter known as the epoch. The number of epochs can be any integer value that lies between 1 to infinity. Traditionally, the ML/ DL model uses large values of epochs.

TABLE 4. EPOCHS VS. BI-LSTM MODEL ACCURACY

Optimizer = Nadam, batch_size = 50, test_size= 0.30				
SN	**Epochs**	**Accuracy %**	**F1-Score %**	**Prgm Exe time (sec)**
1	**205**	**98.52**	**98.62**	4103.7667
2	100	98.48	98.58	1878.8025
3	125	98.48	98.58	2470.6198
4	150	98.48	98.58	2934.2485
5	175	98.48	98.58	3965.207
6	75	98.46	98.56	1465.5138
7	50	98.38	98.48	942.1289
8	45	98.37	98.47	1002.0923
9	35	98.35	98.46	761.2784
10	25	98.3	98.41	527.5244
11	15	98.13	98.25	322.5288
12	5	97.9	98.03	127.0577

This experiment aims to determine the epochs where the Bi-LSTM model provides the highest accuracy. During this experiment, Bi-LSTM hyperparameters were chosen randomly same as in the previous experiment. The larger epochs take a longer time to train the model. We chose epochs ranging from 5 to 205 with some intervals; the accuracy and f1-score are higher for 205 epochs. The training time for Bi-LSTM is increased for a large value of epoch. During this experiment, a batch size of 205 improves the Bi-LSTM model's accuracy of 98.52% in detecting network anomalies.

E. *Experiment-5 Bi-LSTM layers parameters Vs. Accuracy*

We investigated the optimizer, epochs, batch size, and train test data split ratio from the above experiments A-D and found that the value of the hyperparameter: Nadam optimizer, 205 epochs, 50 batch size, 30% test data, and 70% train data generate the best performance which is measured using performance evaluation metrics. During this experiment, we examined the combination of the numbers of units for the multilayer Bi-LSTM model. The output layer provides the probability of selecting either a normal or abnormal class, so the softmax activation function works best for binary class classification problems.

TABLE 5. BI-LSTM LAYERS PARAMETERS VS. ACCURACY

Optimizer = Nadam, batch_size = 50, test_size= 0.30 [Units (activation fn)]				
SN	**Input Layer**	**Hidden Layer 1**	**Hidden Layer 2**	**Accuracy**
1	64 (relu)	50 (relu)	50 (relu)	98.52
2	80 (relu)	64 (relu)	64 (relu)	98.48
3	49 (sigmoid)	128 (Sigmoid)	128 (sigmoid)	98.18
4	16 (selu)	16 (selu)	16 (selu)	97.97
5	16 (relu)	16 (relu)	16 (relu)	97.93
6	4 (relu)	4 (relu)	4 (relu)	97.55
7	8 (relu)	8 (relu)	8 (relu)	97.48
8	4 (sigmoid)	4 (sigmoid)	4 (sigmoid)	97.05

The number of combinations of Bi-LSTM units and activation functions was used in input and hidden layers during this experiment; some of the experiment results are included in Table 4. The experimental result shows that the 64 Bi-LSTM units in the input layer and 50 Bi-LSTM units in both hidden layers produce the highest accuracy of 98.52% during network anomaly detection.

V. CONCLUSION

Comparing our result with existing research [15] for Bi-LSTM, our model produces higher accuracy of 98.52%, which is greater than 94.26%. The values of Bi-LSTM model hyperparameters, including optimizer, epochs, batch size, and the training testing dataset ratio for the multilayer Bi-LSTM neuron architecture (layers, activation function, and memory units) are investigated for the highest detecting accuracy. All the above experimental results show that the Bi-LSTM model with those investigated parameters can effectively improve the detection accuracy and f1-score.

ACKNOWLEDGMENT

The work is supported in part by funding from the NSF awards #1910868, #2219611, and the NSF CyberCorps SFS program.

REFERENCES

[1] S. Samonas and D. Coss, "The CIA strikes back: Redefining confidentiality, integrity and availability in security.," *Journal of Information System Security,* vol. 10, 2014.

[2] N. Moustafa, J. Hu and J. Slay, "A holistic review of network anomaly detection systems: A comprehensive survey," *Journal of Network and Computer Applications,* vol. 128, p. 33–55, 2019.

[3] Y. Fu, Y. Du, Z. Cao, Q. Li and W. Xiang, "A Deep Learning Model for Network Intrusion Detection with Imbalanced Data," *Electronics,* vol. 11, p. 898, 2022.

[4] K. Jiang, W. Wang, A. Wang and H. Wu, "Network intrusion detection combined hybrid sampling with deep hierarchical network," *IEEE Access,* vol. 8, p. 32464–32476, 2020.

[5] W. Xu, J. Jang-Jaccard, T. Liu, F. Sabrina and J. Kwak, "Improved Bidirectional GAN-Based Approach for Network Intrusion Detection Using One-Class Classifier," *Computers,* vol. 11, p. 85, 2022.

[6] L. Vu and Q. U. Nguyen, "Handling imbalanced data in intrusion detection systems using generative adversarial networks," *Journal on Information Technologies & Communications,* vol. 2020, p. 1–13, 2020.

[7] T. Acharya, I. Khatri, A. Annamalai and M. F. Chouikha, "Efficacy of Heterogeneous Ensemble Assisted Machine Learning Model for Binary and Multi-Class Network Intrusion Detection," in *2021 IEEE International Conference on Automatic Control & Intelligent Systems (I2CACIS)*, 2021.

[8] T. Acharya, I. Khatri, A. Annamalai and M. F. Chouikha, "Efficacy of Machine Learning-Based Classifiers for Binary and Multi-Class Network Intrusion Detection," in *2021 IEEE International Conference on Automatic Control & Intelligent Systems (I2CACIS)*, 2021.

[9] C. Yin, Y. Zhu, J. Fei and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *Ieee Access,* vol. 5, p. 21954–21961, 2017.

[10] Z. Chen, C. K. Yeo, B. S. Lee and C. T. Lau, "Autoencoder-based network anomaly detection," in *2018 Wireless telecommunications symposium (WTS)*, 2018.

[11] M. Ganesh, A. Kumar and V. Pattabiraman, "Autoencoder Based Network Anomaly Detection," in *2020 IEEE International Conference on Technology, Engineering, Management for Societal impact using Marketing, Entrepreneurship and Talent (TEMSMET)*, 2020.

[12] W. Xu, J. Jang-Jaccard, A. Singh, Y. Wei and F. Sabrina, "Improving performance of autoencoder-based network anomaly detection on nsl-kdd dataset," *IEEE Access,* vol. 9, p. 140136–140146, 2021.

[13] J. Gao, "Network Intrusion Detection Method Combining CNN and BiLSTM in Cloud Computing Environment," *Computational Intelligence and Neuroscience,* vol. 2022, 2022.

[14] A. G. Salman, Y. Heryadi, E. Abdurahman and W. Suparta, "Single layer & multi-layer long short-term memory (LSTM) model with intermediate variables for weather forecasting," *Procedia Computer Science,* vol. 135, p. 89–98, 2018.

[15] T. S. Pooja and P. Shrinivasacharya, "Evaluating neural networks using Bi-Directional LSTM for network IDS (intrusion detection systems) in cyber security," *Global Transitions Proceedings,* vol. 2, p. 448–454, 2021.

[16] Y. Imrana, Y. Xiang, L. Ali and Z. Abdul-Rauf, "A bidirectional LSTM deep learning approach for intrusion detection," *Expert Systems with Applications,* vol. 185, p. 115524, 2021.

[17] I. Kandel and M. Castelli, "The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset," *ICT express,* vol. 6, p. 312–315, 2020.

[18] M. Tavallaee, E. Bagheri, W. Lu and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE symposium on computational intelligence for security and defense applications*, 2009.